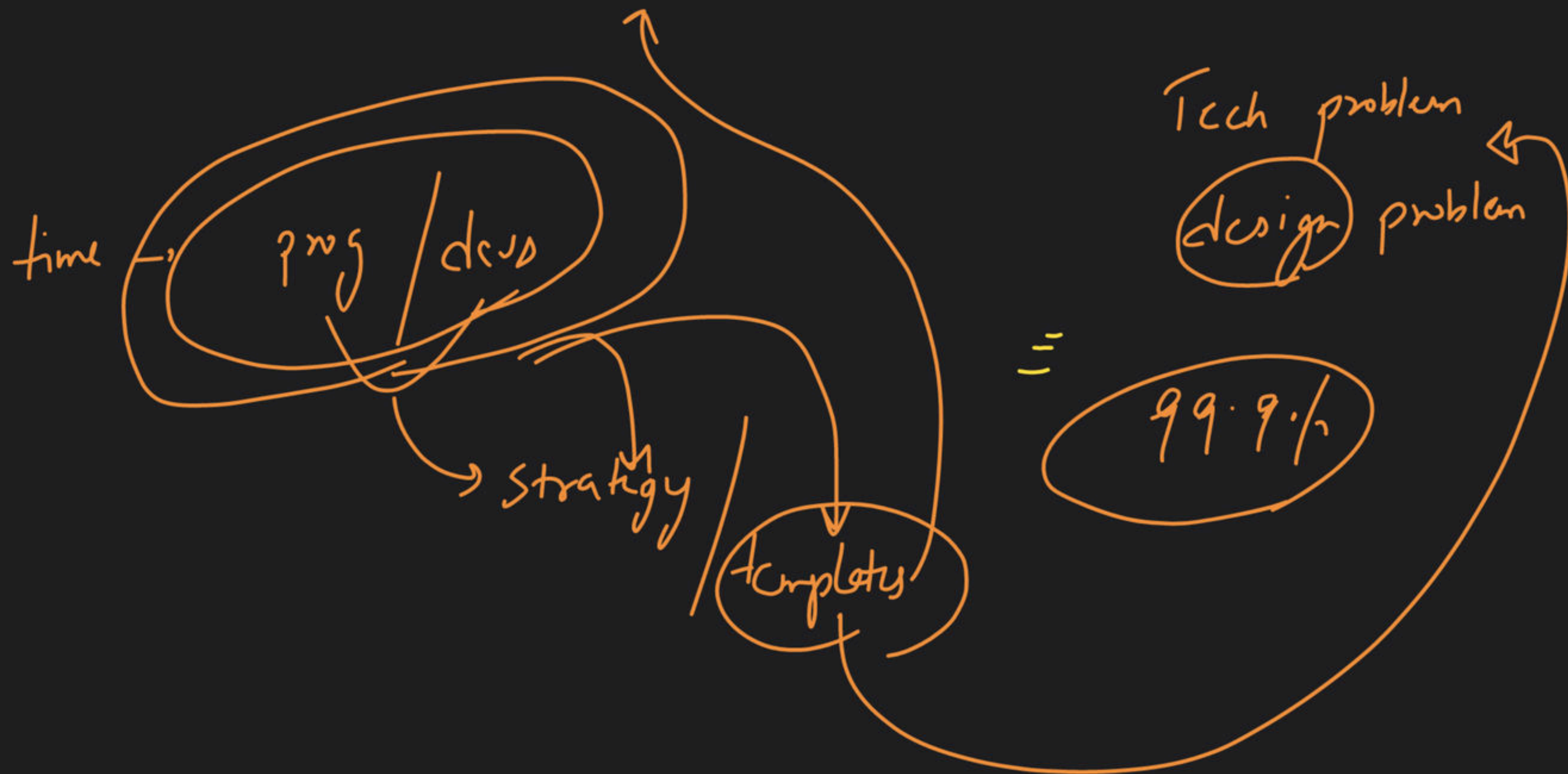
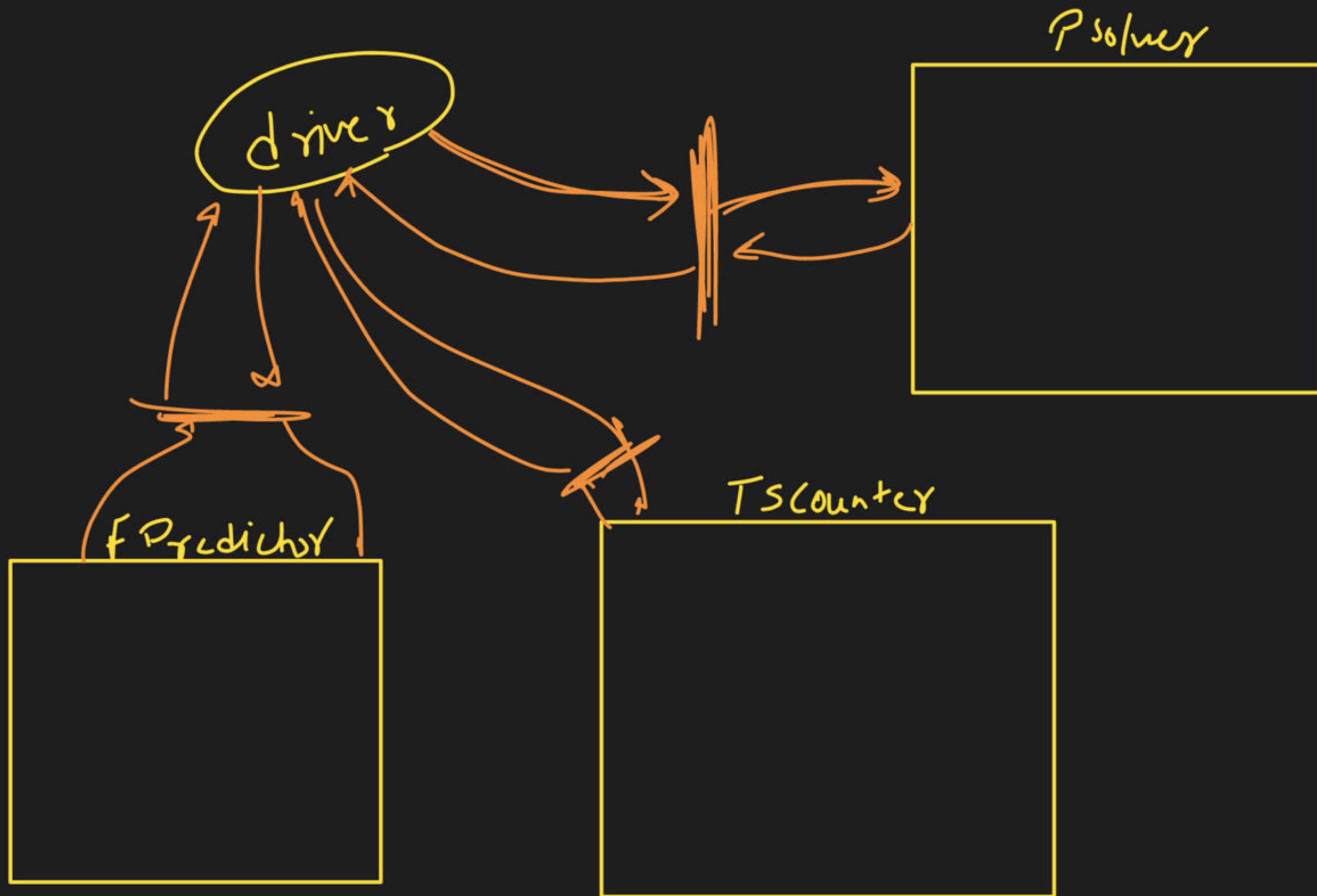


Introduction to Design Patterns, Factory & Abstract Factory Pattern

Special class

→ what is Design Pattern?





Babbar → CodeHelp → Article →

template

Title → < 500

desc → < 3000

ex → 2

ex → 2

10 → approval

approval

→ Car

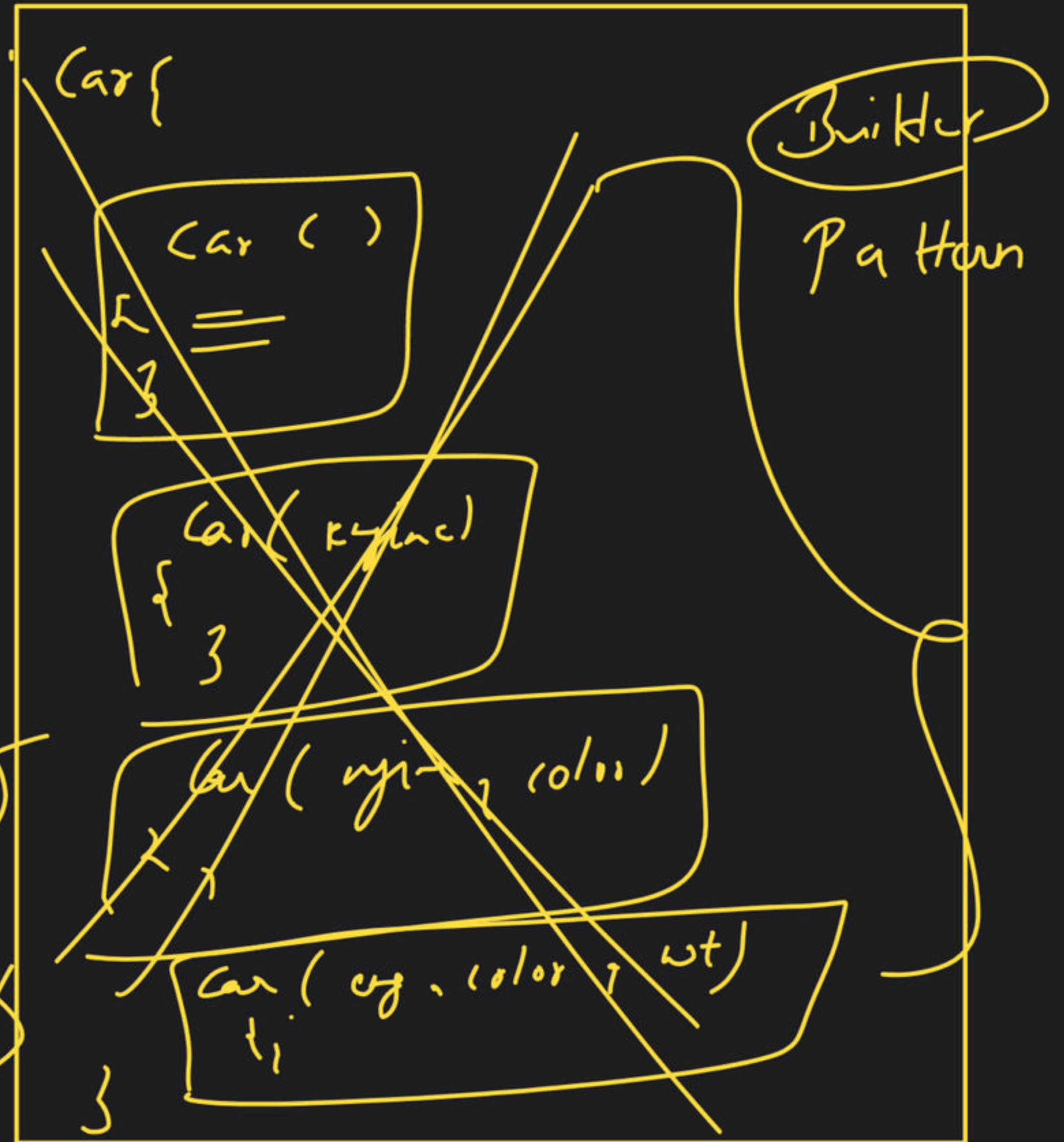
→ Car (engine)

→ Car (engine, color)

→ Car (eng, color, wt)

Telescoping pattern

ANTI-PATTERN



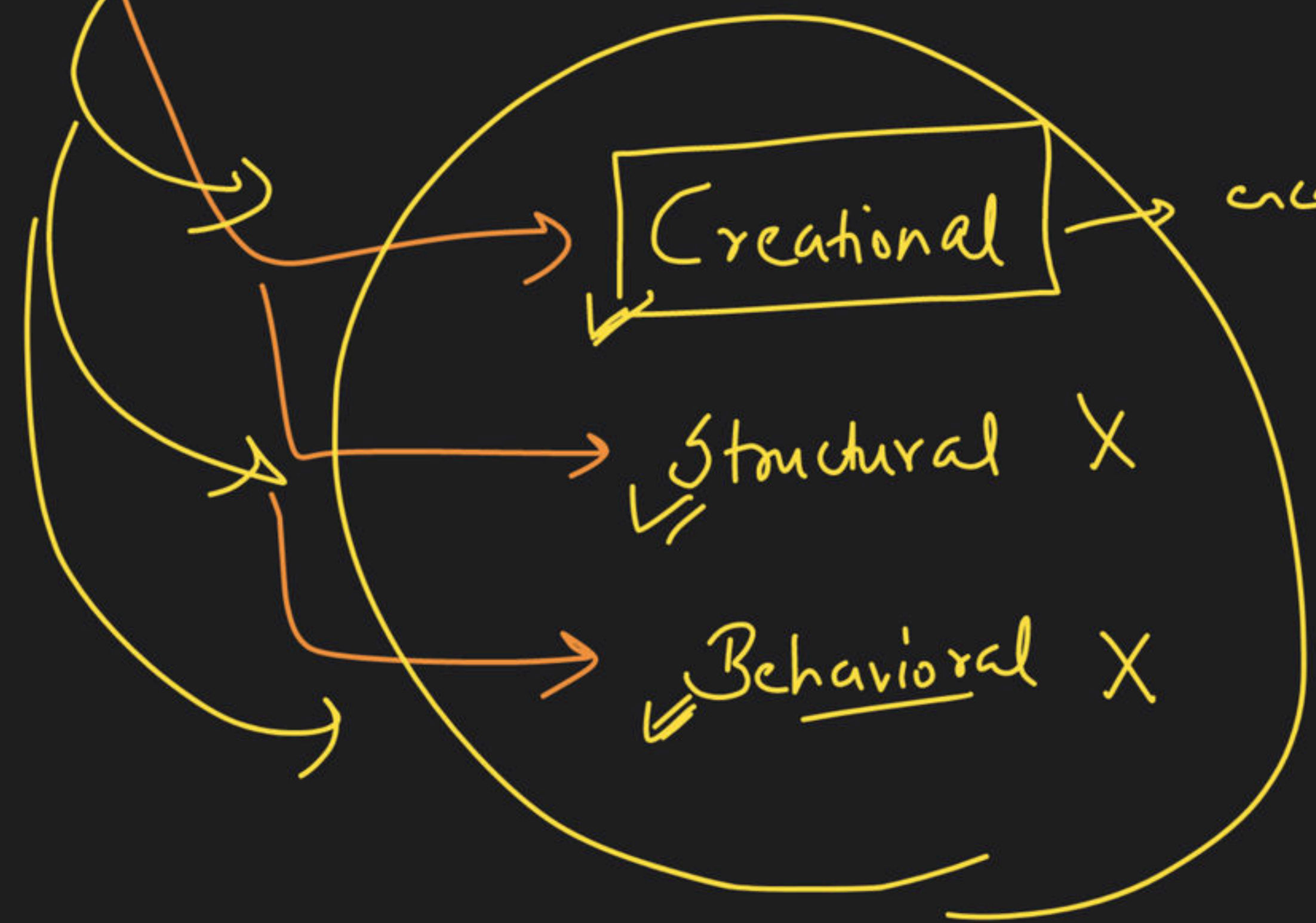
Why Design Pattern?

→ didn't want to
reinvent the wheel

→ Categorisation:-

GOF

4 log



encapsulate Obj-Creation process

factory method

Calendar.getInstance()

→ NumberFormat.getInstance()

Example → Abstract factory

→ DocumentBuilderFactory → newInstance()

→ TransformerFactory → newInstance()

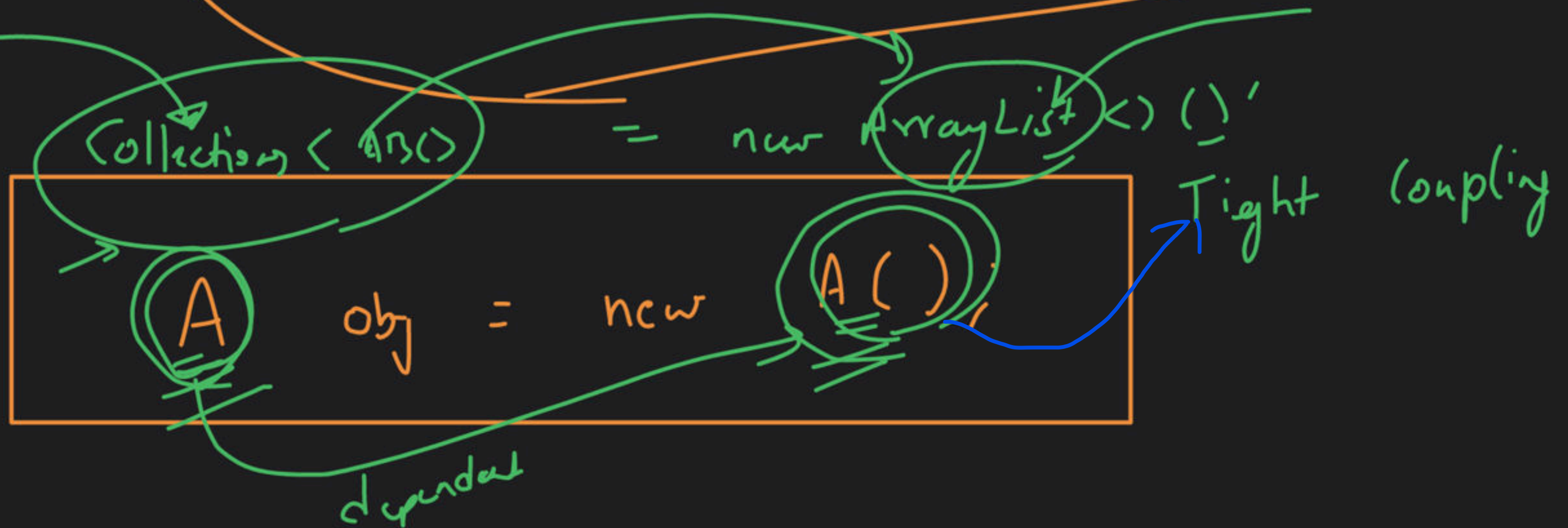
① Factory Method Pattern:-



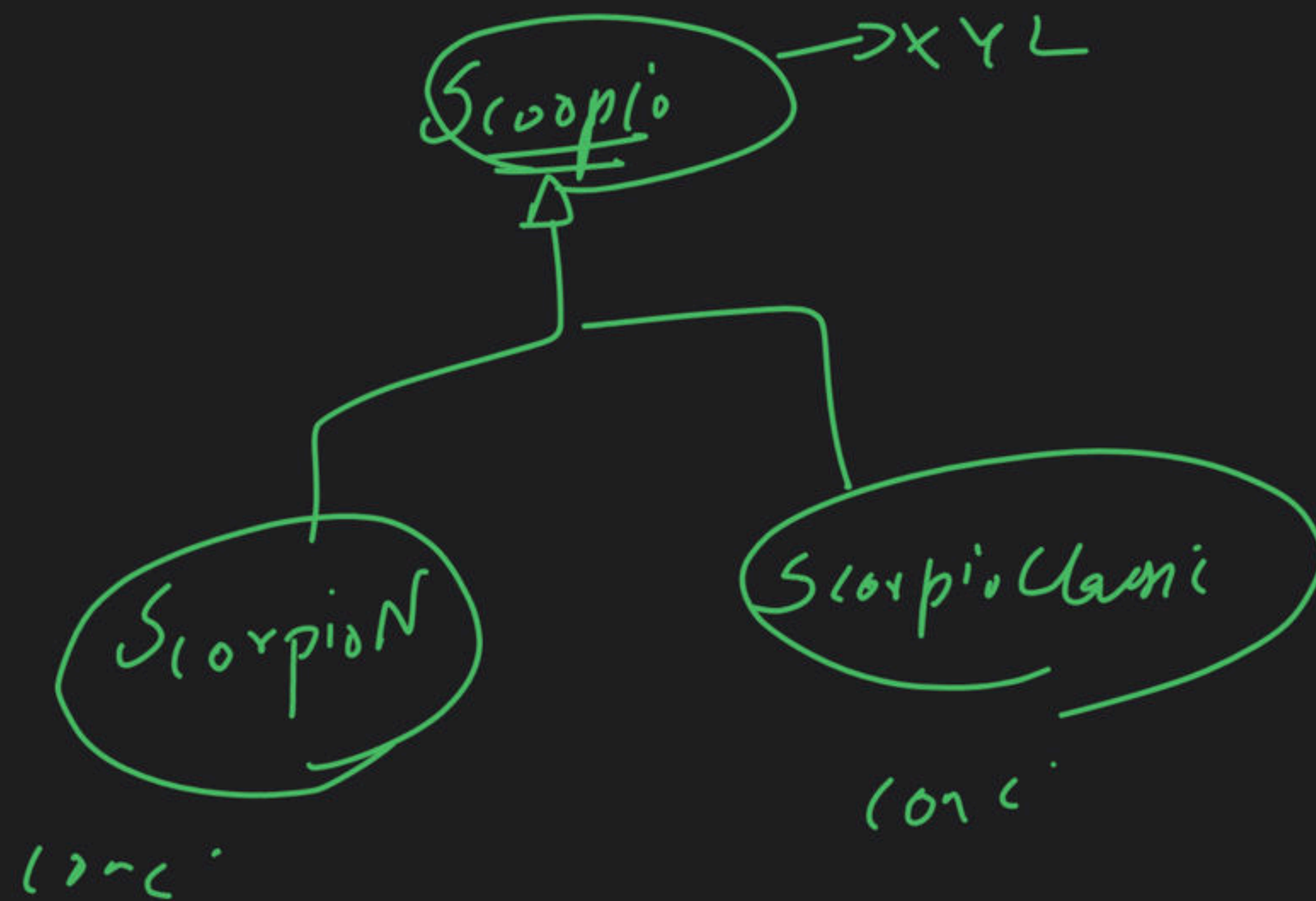
It provides an Interface for obj creation
but delegating the actual instantiation
of objects to subclasses

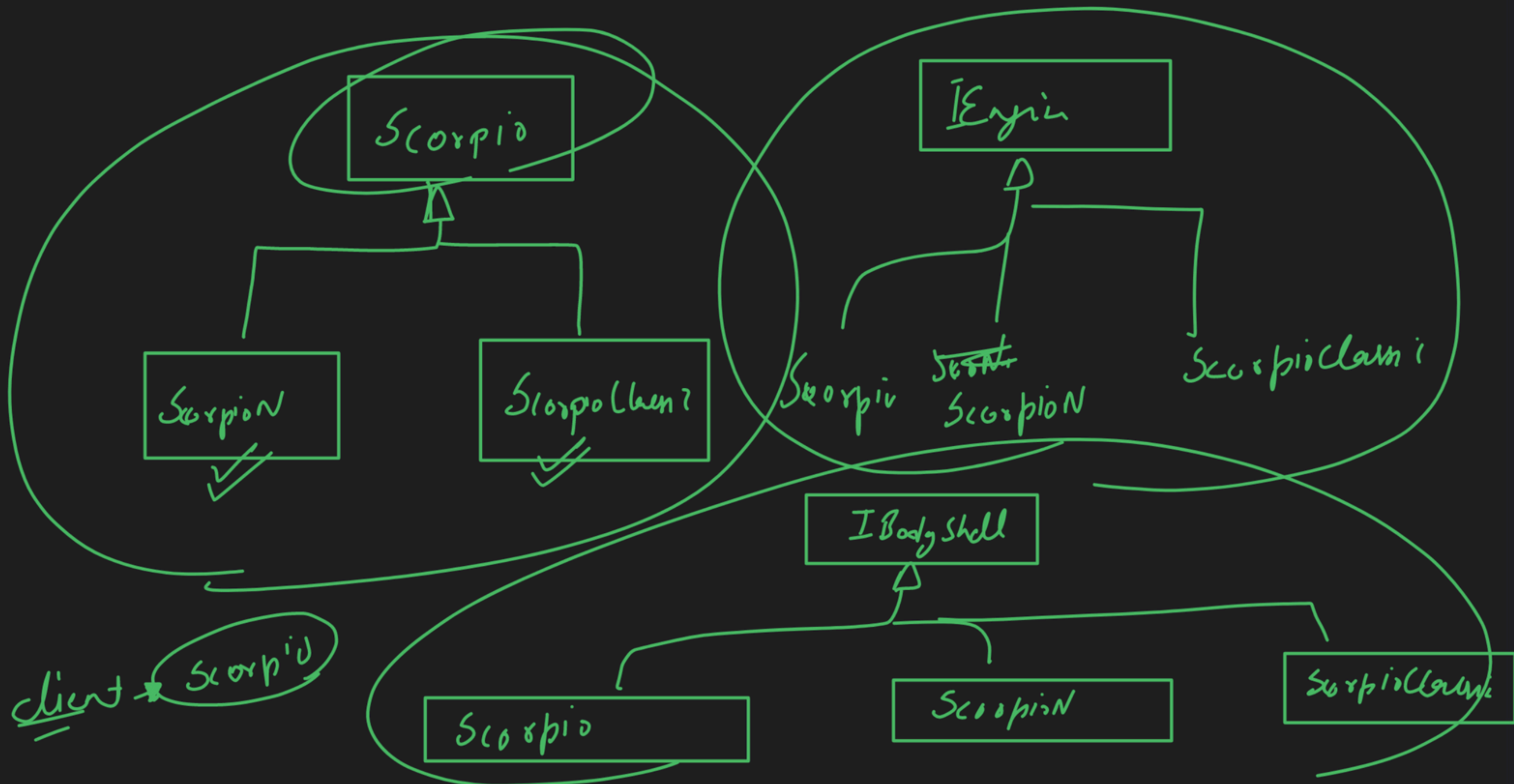
2 min
Break

Ex

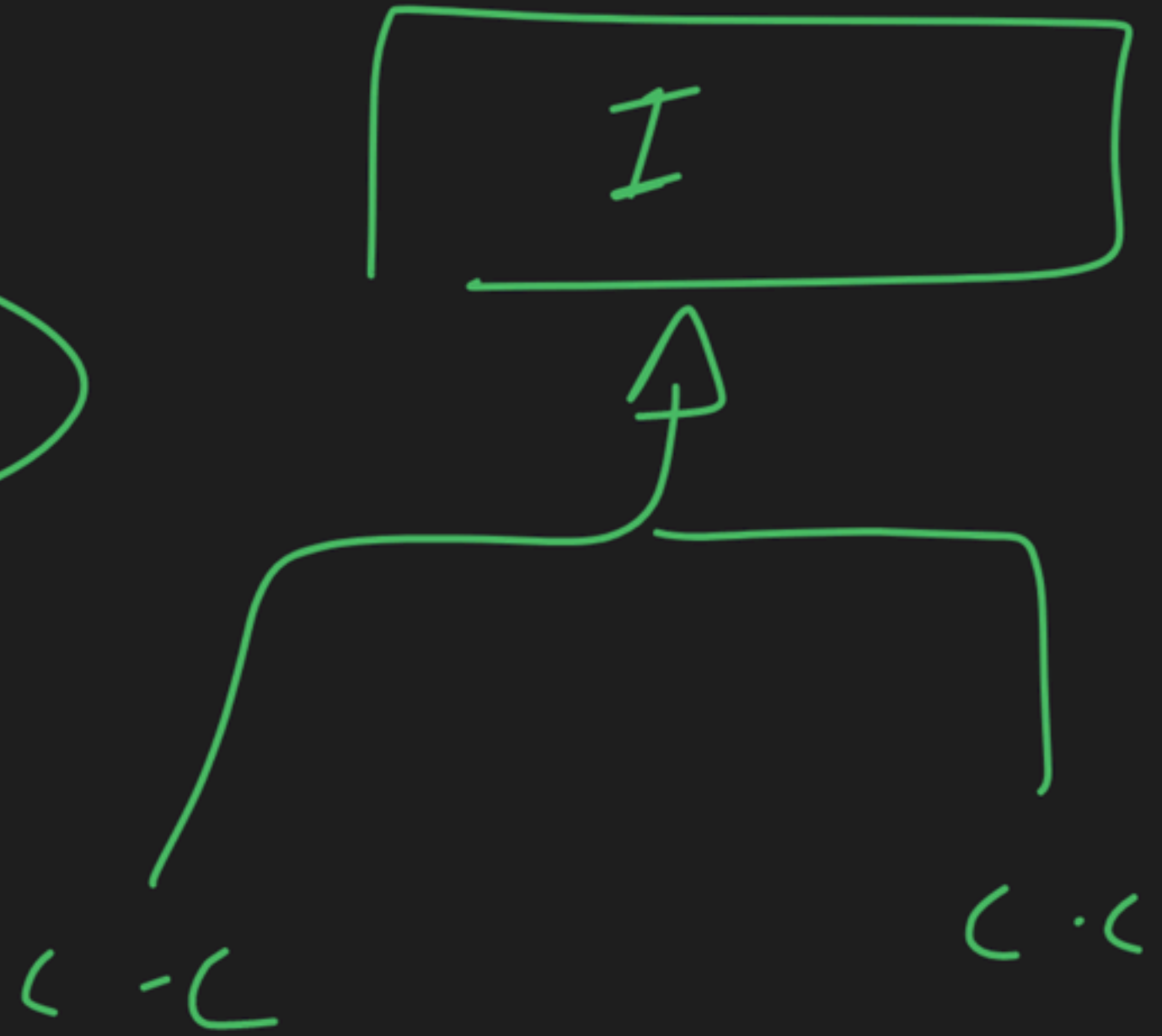


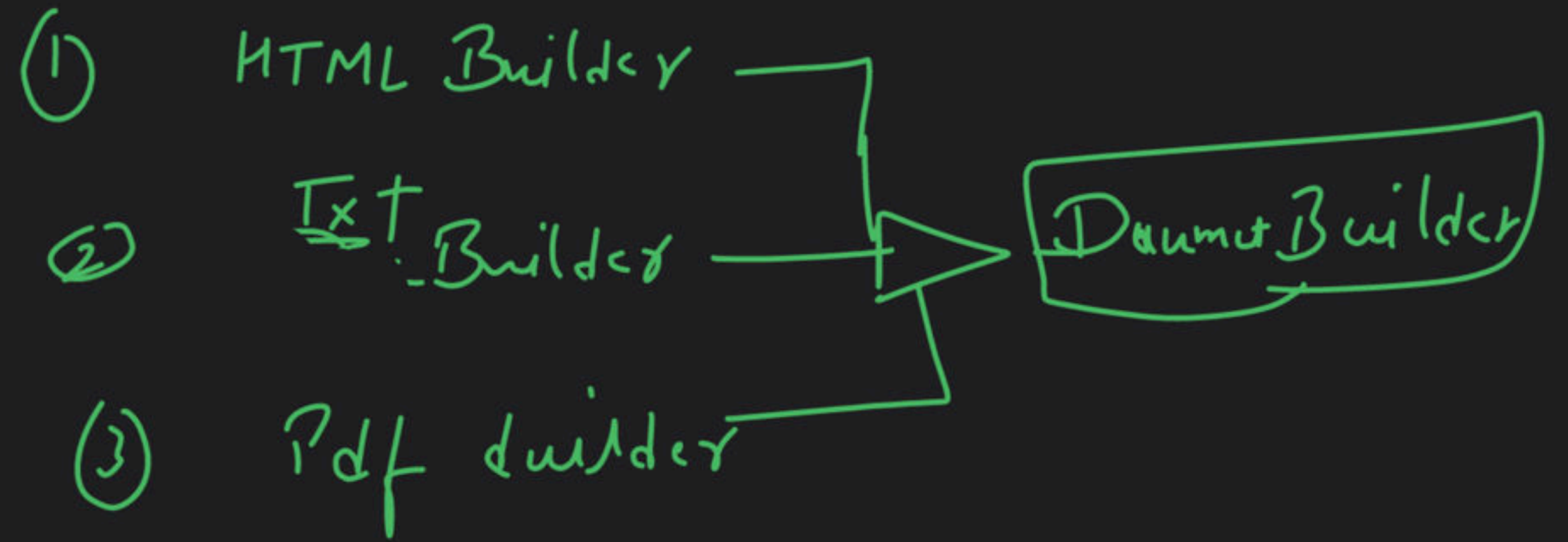
always code to an Interface, not an Implementation"



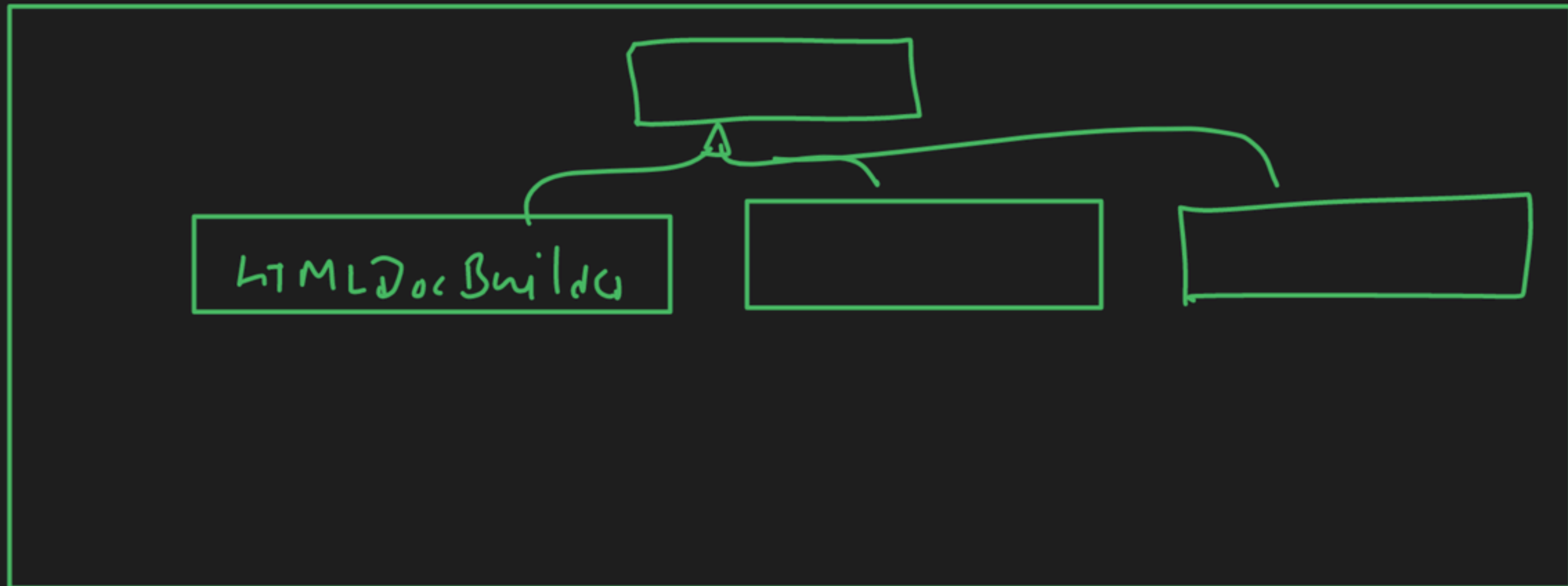


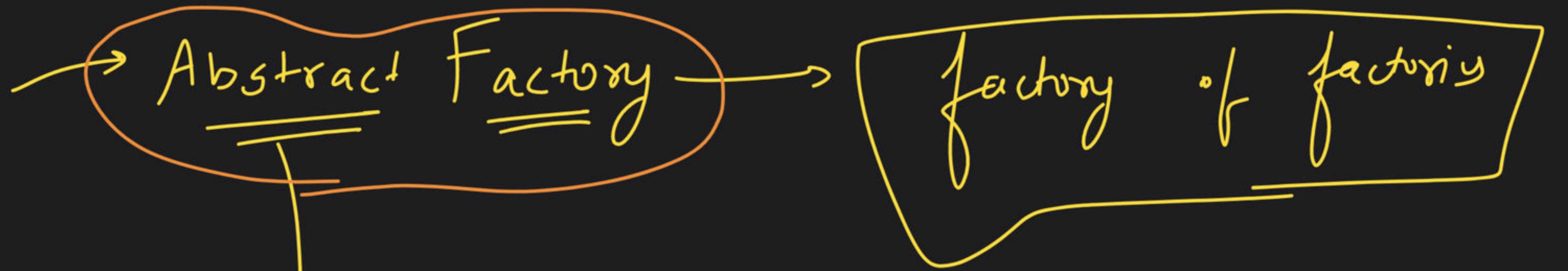
If obj = new CC()



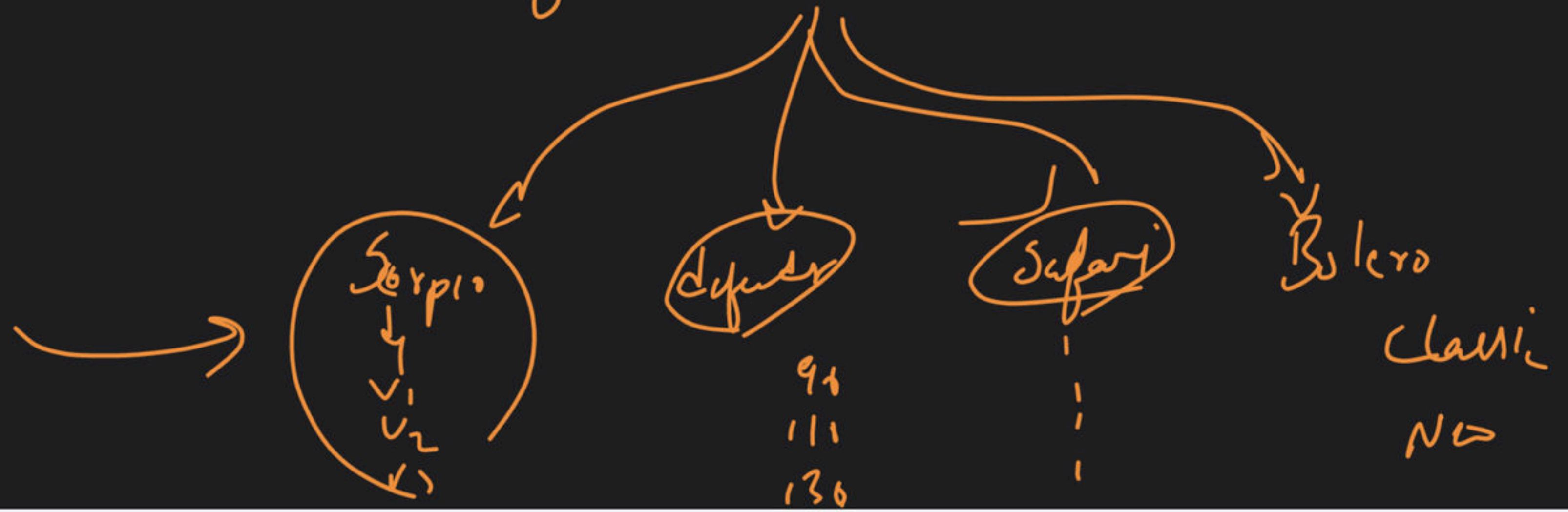


ITEM → Var1 → Var2 → Var3

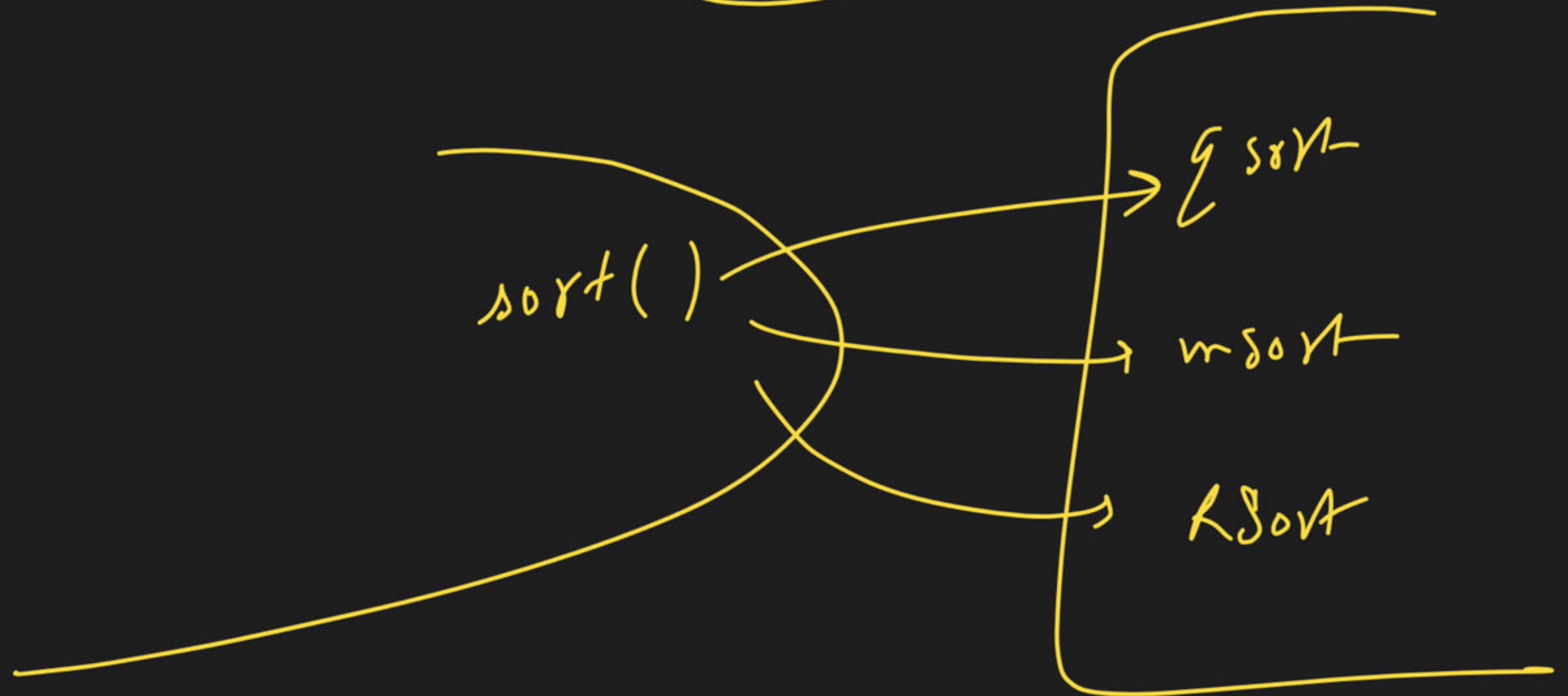


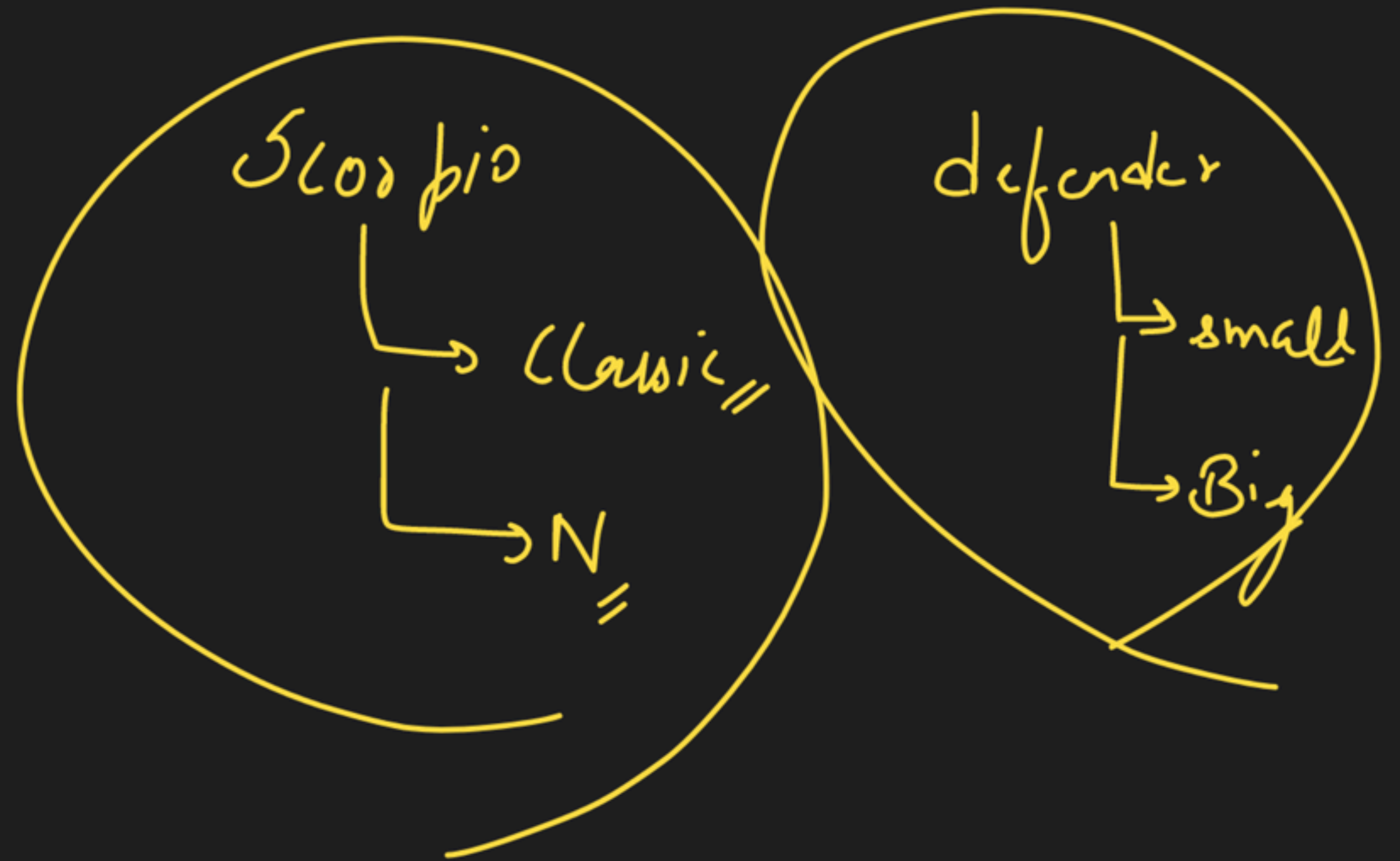


Solves the problem of creating family
of related products



⇒ defining an interface to create families of
dependent objects without concrete class





Factory Method

→ Scorpion
↳ (F.M)

→ Scorpion(LamC
↳ F.M)

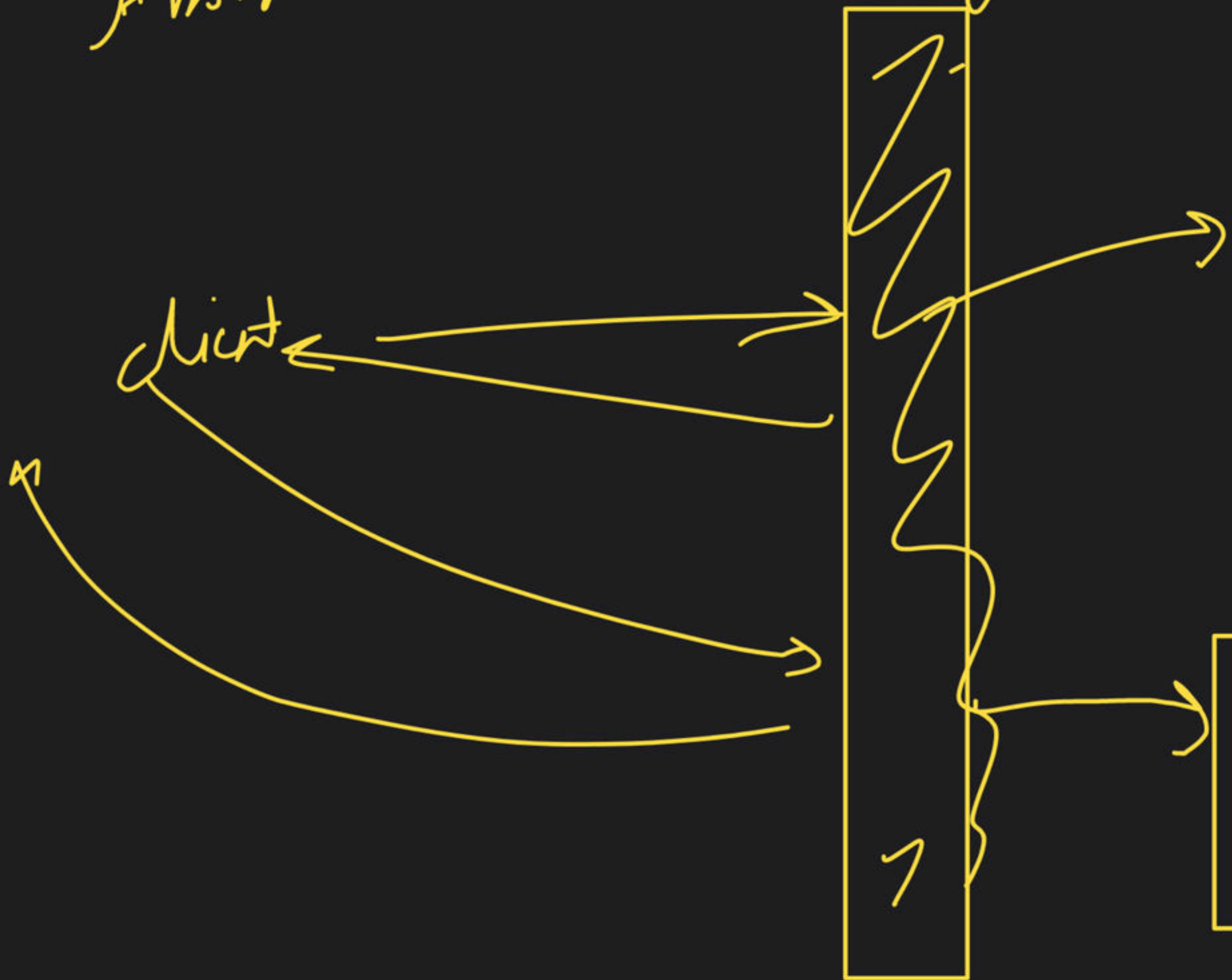
→ Simple Factory

Abstract Factory

abstract
layer

Scorpio
factory

Defender
Factory



Use Case

